VOLUME III, NO. 6 -:- JUNE 1981

OSIO Newsletter

Unravl: A Strategy

By Brian Goodhart

Have you ever tried to learn how someone else's BASIC program works? Have you masochists out there ever tried to unravel a BASIC program written by OSI?

Here is a tool for programmers using BASIC under OS-65U. Nothing fancy, just a merger of two separate 65U features.

The 65U manual describes the FLAG commands on page 24. FLAG 7 enables a BASIC statement trace: invoke it, and the system will display each line of code on the terminal when it interprets it. You can distinguish program from output by the asterisk which precedes traced program lines.

Also in the 65U manual, on page 48, we see that POKE 11686,17 enables simultaneous output to the terminal and line printer (POKE 11686,129 for a serial printer).

Both of these features are probably old hat for you. But if you use FLAG 7, do you sit there and hammer Control S to stop the display and see what is going on? Do you take copious notes to record what you learn from the TRACE? Do you get as frustrated as I did?

There's a better way. Combine these two features, and what do you get? Hard copy of your statement trace! Now you can read a program listing that shows every line of code as it is executed. If Line 33200 says "GOTO 157" the next line in the listing is Line 157. I can attest that this feature is very useful when you have to determine how another author's program works.

Example 1 shows how the POKE and FLAG entries can clear up an ON - GOTO.

EXAMPLE 1

```
 5 FLAG 7: POKE 11686,129
 10 PRINT "HELLO THERE."
 20 FOR I = 1 TO 5
 30 ON I GOTO 40,60,80,100,120
 40 X = I
 50 GOTO 150
 60 X = X ^ 2
 70 GOTO 150
```

continued on page 2

Disk Copy: Single Drive

By Jeffrey Krauss

I knew that one of the first things I would want to do after my C4P-MF arrived was to make backup copies of the system software and other programs I ordered. But I couldn't figure out from the sales literature how to copy a disk with only a single disk drive. No problem, they said, simply copy track by track and sector by sector.

Well, in fact that is not trivial by any means, and OSI supplies no documentation to describe the simple tricks that are needed to do a single-drive copy. Therefore, this description may be helpful to new OSI users in the future, so that it won't take them as long as it took me to figure this out.

The general procedure is as follows: first, initialize the blank disk. Second, copy track 00 from the operating system onto the initialized disk. Third, copy the remainder of the operating system, tracks 01-13, onto the initialized disk. Finally, copy the other files onto the new disk.

Initializing a disk is simple and straightforward, but be sure you insert a disk that is either blank or expendable, because it wipes out everything that was previously on the disk. The second and third steps have other little pitfalls for the unwary.

Copying track 00 requires the use of the Track 0 Read-Write Utility, which is poorly documented in the OSI Users Manual. This program is found on track 13 of the system disk, and can be loaded with the following command: DISK!"CALL 0200=13,1". Respond to the menu with a "2" to select the Track 0 utility. With the system disk still loaded, respond R4200 to read track 00 into memory starting at location hex 4200. Then remove the system disk, insert the initialized disk, and respond W4200/0200,8 to load the material starting at location 4200 onto the new disk. The 0200 is the hex memory location that the system jumps to upon a cold start (i.e.,

continued on page 2

Unravl: A Strategy

continued from page 1

```
80 X = X / 4
90 GOTO 150
100 X = X - I
110 GOTO 150
120 X = X + I - 3
150 NEXT I
160 PRINT " X= ";X
200 END
```

Of course, this tactic should be used judiciously. Before you put "POKE 11686,17:FLAG 7" at the beginning of all your programs, save yourself some paper and time. Read through the program and find all the FOR-NEXT loops. Unless you own Diamond Paper, you don't want a hard copy trace of every execution of every loop. Start and stop the trace and the printout to study the parts that are hard to follow.

Suppose you're going to operate on a variable in a loop that will execute 100 times, and you'd like to know what the variable is at the beginning and at the end of the loop. Example 2 shows how you can do it:

EXAMPLE 2

```
 90 POKE 11686,129:FLAG 7
 100 X=4 * RND(1)
 110 FOR I = 1 TO 100
 120 X=INT(X+2.345)
 130 IF I<3 THEN POKE 11686,129:PRINT
"I= ";I,"X= ";X
 135 IF I=3 THEN POKE 11686,1:FLAG 8
 140 IF I>98 THEN FLAG 7:POKE 11686,129:
PRINT"I= ";I,"X= ";X
 150 NEXT
```

Disk Copy: Single Drive

continued from page 1

hitting "BREAK" and then "D"), and the "8" indicates 8 pages.

At this point you are no longer in BASIC but rather in the kernel of the Operating System, and I have always found it most convenient to load the system disk back in, hit the "BREAK" key, and re-boot.

Copying Tracks 01-13 requires the use of the CALL and SAVE commands. Most of these tracks contain a single file, eight pages long, starting at sector 1. A track containing a single file can be copied with a single pass: with the system disk loaded, type DISK!"CALL 4200=01,1"; then remove the system disk, insert the new disk, and enter DISK!"SAVE 01,1=4200/8". However, tracks 06, 12, and possibly others (who knows?) have several files, and require a separate CALL, SAVE for each file. Here it is useful to run the SECDIR utility program that is supplied on the system disk to determine what is on each track. If these tracks are not copied properly, the disk will not boot in and the directory will not be stored properly.

The remainder of the disk can be copied using the LOAD and PUT commands, using file names determined by using the DIR program. For example, the program CREATE can be copied as follows: with the system loaded, type DISK!"LOAD CREATE"; then, with the new disk loaded, type DISK!"PUT CREATE". The system will use the directory information from track 12 to determine which track the file should be loaded into.

Of course, this approach is a bit tedious, requiring the continual swapping of disks. It might be more efficient to buy from a vendor a disk copy routine that will accomplish the copy with only four or five swaps. Surprisingly, Ohio Scientific itself does not offer such a program, but there are several vendors who offer the software for about $15-20.

Nevertheless, the procedure outlined here works properly; the keys are knowing how to use the Track 0 Read/Write utility, and running SECDIR to determine the layout of each track. Unfortunately, the OSI User Manual doesn't provide much guidance in this area.

OS-65D Notes

OS-65D UTILITIES

Examining the utilities not only will help us understand how a number of PEEKs and POKEs can be used, but can also show us how we can select what we need from the various utilities and combine them into a single program. This is very worthwhile if you have a single disk system and need the utilities on nearly every disk.

It will be easier to follow the discussion if you have a copy of the listing of the utilities handy. One place you can find them is in the final issue of The Small Systems Journal which consists of eight sections. Section 8 is a listing of the Utilities.

The CHANGE Utility

The program immediately drops down to line 1000 and starts by adjustment of the (printer) line width. The width is changed to anywhere between 14 and 255 by poking the desired width into decimal 23. The total number of characters for all the fields is then poked into 24. There are 14 characters per field, so that this number must be an integral number times 14 or data on two consecutive printed lines will not line up.

From line 1100 and onward, we see that the number of pages of available memory can be determined by peeking at decimal 8960, and that the workspace can be decreased by poking into 8960 a smaller number of pages than what is actually available in memory. The space that is made unavailable to BASIC in this way can be used for other purposes.

We then go back up to line 36 where we are given other opportunities to change BASIC's workspace.

The buffers are set up by the operating system at the bottom of BASIC's workspace where they do not affect BASIC unless you try to bring in data from the disk. If you do try to bring in data, it will overwrite the portion of the BASIC program residing in the buffer area with disasterous results. Consequently, in order to prevent this, the buffers must not be in the BASIC workspace if data is to be moved between BASIC and the disk. The CHANGE utility allows you to move the lower bound of BASIC up above the area occupied by the buffers. The other option open to you is to lower the upper bound of BASIC and put the buffer(s) on top.

Lines 40 onward ask you how many buffers you want to use and whether you want to leave room below BASIC for something in addition to the buffers. You can leave additional space below BASIC for some data, for a machine program, or possibly other purposes.

It is important to realize that everything that resides in the region between BASIC's original lower bound and the new lower bound at the time of disk transfer is saved on the disk. This means that if you put a machine language program or some data between the buffers and BASIC, it will be there the next time you load the program from the disk.

In general you will not want to save the material that happens to be in the buffers when you PUT to the disk, however, you don't have any choice if you have left your buffers on the bottom. If you use buffer #6 on the bottom, one track of the disk will be wasted. If you use #6 & #7, or if you use #7 alone, two tracks will be wasted.

On the other hand, even though you place the buffers on the top, you may want to leave some space on the bottom for some data or a machine program that you want to save along with the BASIC program.

Glancing down to line 390, we note that the lower bound of BASIC is lifted by poking the new bottom address into 120 and 121. The new address is calculated by adding 3072 bytes for each buffer plus as many bytes as are desired for the additional space, and adding the result to the address of the original bottom.

The upper limit, which we may already have lowered by changing the number of pages in 8960, may be lowered further by poking the new upper bound into 132 and 133. Of course, it is not necessary to use 8960 at all if you are going to use 132 and 133, and vice versa; however, 8960 allows only changes in terms of pages, since only the high byte is used.

Next month, continuing our study of OS-65D utilities, we'll give a good deal of attention to CREATE.

JOE KIRSHNER

Flash Card Program

by Stewart C. Meyers

This is a rather basic program with no bells and whistles. It does not ask for your name or do any stroking. It simply flashes a problem and allows you to answer it only for the time it is displayed.

This time is selectable as is the type of problem. After ten problems your score is totaled and you are asked if you want to continue. The missed problems are displayed.

The key to the program is that it runs in "real time", it does not stop for inputs but rather peeks the keyboard while in the timing loop.

This routine is adaptable to other programs. You could for instance scan the entire keyboard not just the numbers and rub-out key. I don't claim this is the optimum code but it works well and fast enough. If you think of your self as a math wizz try it at 50 rather than 9 for speed.

The technique of drawing random numbers while waiting for a shift to be pressed is also note worthy. Even with OSI's short random cycle you don't repeat the problem sequence. The problems are also checked to eliminate repeats which legitimately occur in a true random sequence.

This is written for a C1P with 8K and a 24 x24 display, it can be easily modified to run on other polled key board machines. The X=USR(X) is a screen erase. Substitute your favorite.

```
 100 PRINT"***********************
*
 110 PRINT"  FLASH CARD PROGRAM":
  PRINT:PRINT"*********************
***
 120 PRINT:PRINT"WITH THREE TYPES
OF":PRINT" PROBLEMS....
 130 PRINT"(1) ADDITION FACTS TO 2
0
 140 PRINT"(2) SUBTRACTION FACTS T
0  20
 150 PRINT"(3) MULTIPLICATION FACT
S  TO 12
 160 INPUT"WHICH DO YOU WANT";S
 170 IFS<1ORS>3THEN160
 180 PRINT"A PROBLEM WILL FLASH ON
  THE SCREEN
 190 PRINT"YOU MUST TYPE THE ANSWE
R  BEFORE IT DISSAPEARS
 200 PRINT"YOU MAY ERASE ERRORS
 WITH THE RUB-OUT KEY.
 210 PRINT"IF YOU PRESS RETURN THE
 ANSWER WILL BE CHECKED  IMMEADIAT
ELY
 220 PRINT:PRINT"HOW FAST DO YOU W
ANT TO GO ?"
 230 INPUT"TYPE 1(SLOW) TO 9(FAST)
";TK
 240 IFTK<1THEN230
 250 TK=250/TK
 260 PRINT"PRESS EITHER SHIFT KEY
 TO START
 270 IFPEEK(57088)=254THENX=RND(8)
:GOTO270
 280 X=USR(X)
 290 TD=TK*50
 300 N=0:EN=0:PN=10:E1=0
 310 DIME$(99),Q$(99),B(99),C(99),
D(99)
 320 ONSGOTO330,330,590
 330 B=INT(20*RND(8))
 340 C=INT(20*RND(8))
 350 IFC=0ORB=CTHEN330
 360 ON S GOTO370 ,390,590
 370 IFC>BORB=0THEN330
 380 GOTO400
 390 IFC<BORB=0THEN330
 400 D=B-C:REM C+D=B
 410 FOR H=0TON
 420 IFB=B(H)ANDC=C(H)THEN320
 430 NEXTH
 440 ONSGOTO450 ,470,490
 450 Q$=STR$(C)+"+"+STR$(D)+"="
 460 GOTO500
 470 Q$=STR$(C)+STR$(D)+"="
 480 GOTO500
 490 Q$=STR$(C)+" X"+STR$(D)+"="
 500 B(N)=B:C(N)=C
 510 N=N+1
 520 Q$(N)=Q$
 530 PRINTQ$
 540 B$=STR$(B)
```

OSIO Program Exchange Library

DISKETTE : OS-65D3 NO. 1 OCTOBER 1, 1980

THE APPOINTMENT PROGRAM IN THE LIBRARY USES THE FOLLOWING PROGRAMS : 'APPT', 'APPTAD', 'APPTED', APPTFI', 'APPIN', 'CALEND', 'INSTR1', AND 'INSTR2'. IT ALSO HAS SAMPLE DATA FILES KNOWN BY : 'DATAP1' AND 'DATAP2'. THESE DATA FILES OR SOME OTHER DATA FILES WITH THE SAME NAME ARE REQUIRED.

THE PROGRAM 'FIND&P' IS USED TO COPY DATA FILES.

'BIO' IS A BIORHYTHYM PROGRAM WRITTEN FOR THE C2-8P WITH VIDEO DISPLAY.

'NUMCON' IS A HEXIDECIMAL NUMBER TO DECIMAL NUMBER CONVERSION PROGRAM

'EDIT' IS A TEXT EDITOR/FORMATTER WRITTEN IN BASIC. IT REQUIRES 48K TO RUN EFFECTIVELY; HOWEVER, THE EDITOR CAN BE SEPARATED FROM THE FORMATTER SO THEY CAN RUN IN A 32K DISK BASED SYSTEM. THIS PROGRAM WAS COPIED OUT OF THE MAY 1979 ISSUE OF KILOBAUD.

'EDITOR' IS A PROGRAM TO CHANGE DATA FILES.

'RENUM' IS A BASIC RENUMBERING PROGRAM TO MAKE YOUR BASIC PROGRAMS LOOK NICE. USE AN INDIRECT FILE TO APPEND THE RENUMBERING PROGRAM TO YOUR BASIC PROGRAM. THEN JUST RUN63900. YOUR PROGRAM WILL BE RENUMBERED

'MOD550' IS AN ASSEMBLY LANGUAGE PROGRAM THAT DOES THE NECESSARY WORK TO INTERFACE A 550 BOARD(X10) TO A MODEM

'PRONLY' IS A UTILITY PROGRAM TO PREPARE A DISK OF SELECTED PROGRAMS FROM ANOTHER DISK AUTOMATICALLY.

'ASMUSR' IS A UTILITY PROGRAM TO CREATE BASIC DATA STATEMENTS OF AN ASSEMBLY LANGUAGE ROUTINE OR PROGRAM.

'LISTER' IS A UTILITY PROGRAM TO AUTOMATICALLY LIST A SERIES OF SPECIFIED PROGRAMS.

'RENUMA' A PROGRAM TO RENUMBER YOUR BASIC PROGRAM ON DISK.

'CMPRON' A COMMENTED PROGRMA OF 'PRONLY'.

'CRENAM' A PROGRAM TO CREATE NAME ON A DIRECTORY.

'MESSAG' IS A PROGRAM TO STORE MESSAGES ON DISK.

'REACT' IS A PROGRAM TO TEST YOUR REACTION TIME. FOR THE C8P OR C4P.

'TELECO' 'TELCO1' & 'TELE#' ARE VARIATIONS OF MOD550 BUT FOR DIFFERENT PORTS. ALL ARE ASSEMBLY CODE.

'MEMTST' A MEMORY TEST PROGRAM THAT IS THOROUGH.

DISKETTE : OS-65U NO. 1 APRIL 25, 1981

BEFORE RUNNING THE PROGRAM 'CKBOOK', CREATE ON ANOTHER DISKETTE 'CRDITS', 'CHRGES', AND 'SUMMRY'.

BEFORE RUNNING 'LABSRT', CREATE ON A DIFFERENT DISK 'LABCFA', 'LABCFC', 'LABCFF', 'LABCFG', 'LABCFM', 'LABCFQ', AND 'LABCFT' (TO RUN WITHOUT CHANGING THE PRESENT PROGRAM). SOURCE DATA MUST BE IN THE OSI LABLE FORMAT -- 6-FIELD RECORD WITH FIELD LENGTHS OF 25, 25, 25, 25, 10, & 10.

DIR	DISPLAYS THE DIRECTORY OF THIS DISK.
CKBOOK	IS A PROGRAM TO RECONCILE YOUR CHECKBOOK THERE ARE INSTRUCTIONS WITH THE PROGRAM.
LABSRT	ALONG WITH 'LABLE2' SORTS MAILING LABELS AS CREATED BY THE LABLE PROGRAM
LANDER	A LUNAR LANDER USING A SERIAL TERMINAL LIKE AN ADM-3 TERMINAL
HEXDEC	CONVERTS HEXIDECIMAL NUMBERS TO DECIMAL NUMBERS
DCRYPT	AN AID TO SOLVING CRYPTOGRAMS. INSTRUCTIONS ARE INCLUDED WITH THE PROGRAM.
HEADER	DUMPS THE HEADER OF A DBMS FILE. IT USES ITS OWN ASSEMBLY LANGUAGE ROUTINE SO IT IS FASTER THAN THE PROGRAM SUPPLIED BY OSI.
MTHQIZ	MATH QUIZ - A VERY GOOD PROGRAM FOR TESTING YOUR MATHEMATICAL SKILLS
DMDUMP	A DUMP PROGRAM FOR DBMS FILES
ZONEX	A PROGRAM TO GENERATE PATTERNS ON YOUR PRINTER
FFUN	ANOTHER PROGRAM TO HAVE FUN WITH
COPIER	THE COPY PROGRAM SUPPLIED WITH OS65U. IT IS USED BY THE 'MAKE' PROGRAM.
MAKE	USED BY THE OSIO LIBRARIAN TO MAKE COPIES OF THE PROGRAM EXCHANGE LIBRARY DISKS. IT MAKES USE OF THE INDIRECT FILES OF OS65U BASIC.
RENUMB	RENUMBERING PROGRAM FOR BASIC PROGRAMS. INSTRUCTIONS ARE FOUND IN LINE NUMBERS 12040 -12110 CLOSE EXAMINATION OF LINE #'S 10000 THRU END OF THE PROGRAM WILL BE OF GREAT ASSISTANCE. PROGRAM DOES NOT EXPAND SPACE BUT GIVES A WARNING MESSAGE.
FORMAT	SKELETON OF OSIO TOOLBOX SAMPLE PROGRAM STRUCTURE
MENU	A SAMPLE MENU PROGRAM FOR RUNNING THE RENUMB PROGRAM
LETTER	PROGRAM TO AID IN THE WRITING OF A LETTER
LISTER	A DATA FILE LIST PROGRAM - PRINTS ALL THE DATA CONTAINED IN A FILE
FMPFMP	FILE MAP OF THE FILE MAP PROGRAM
FILMAP	A VARIABLE MAPPING PROGRAM AS APPEARED IN THE 9/80 MICRO. REQUIRES 2 FILES FOR INPUT. ONE IS A DATA FILE OF THE BASIC PROGRAM (THIS IS DONE BY DOING: LOAD OF THE BASIC FILE, OPEN THE DATAFILE,CHANNEL, AND LIST%CHANNEL, PRINT%CHANNEL,'OK') THE SECOND DATA FILE IS NEEDED FOR RECORDING THE INFORMATION AS IT IS PRODUCED. IT CAN BE PRINTED WITH THE REPORT FUNCTION AT LINE 2000
FWORK	A LITTLE CLEANUP AND MOVING TOOL.
FMOSIO	THE FILMAP PROGRAM IN THE OSIO TOOLBOX SUGGESTED STRUCTURE
PASKEY	PROGRAM TO DECODE THE PASSWORDS ON 65U FILES
DDIR	PROGRAM TO DISPLAY A DIRECTORY IN DOUBLE COLUMNS
KOPIER	PROGRAM TO COPY AN ENTIRE DISK
MEM-IO	PROGRAM THAT ILLUSTRATES SEVERAL TECHNIQUES IN 65U AND BASIC: (1) ALLOCATING MEMORY FOR A SECOND 65U BUFFER, (2) SETTING THE INPUT DISTRIBUTOR FOR MEMORY INPUT, (3) READING 65U DATA FILES VIA USR(X), AND (4) COMBINIG A SET OF ROUTINES TO TEST AND DEBUG I/O CODE BEFORE INSTALLING IN LIVE APPLICATIONS.
ED	AN EDITOR PROGRAM WHICH WAS DERIVED FROM AN OLD ARTICLE IN KILOBAUD MAGAZINE. INSTRUCTIONS FOR USING 'ED' ARE IN THE FILE 'INSTRU'. 'TXPRNT' CAN BE USED TO PRINT THE INSTRUCTIONS.

TXPRNT	PROGRAM TO PRINT FILES CREATED THROUGH THE 'ED' PROGRAM. 'INSTRU' CONTAINS INSTRUCTIONS FOR USING IT.
NEWTEX	PROGRAM TO INITIALIZE A FILE FOR 'ED'.
LSTR	A VERY POWERFUL TOOL TO EXAMINE FILES AND CHANGE THEM IF NEEDED. A COPY OF THIS PROGRAM CAN BE INCLUDED AT THE END OF YOUR PROGRAMS AND JUMPED TO IF '9999'IS A RESPONSE TO ANY INPUT (MUST BE PROGRAMED).
VARLST	WILL LIST ALL VARIABLES USED IN A PROGRAM AND WHERE THEY ARE USED (LINE #).
VARUS1	(A FILE 'VAR%7' IS REQUIRED) BUILDS A NEW PROGRAM AND PUTS IT INTO 'VAR%7' WHICH WILL PRINT VARIABLE NAMES IN FILE #7 EACH TIME A VARIABLE IS USED AS THE PROGRAM IS RUN. FROM THAT YOU CAN ORDER THE VARIABLES SO THE MOST USED VARIABLES ARE FIRST, AND THE LEAST USED ARE LAST, TO SPEED EXECUTION. BE CAREFUL OF VERY LARGE PROGRAM!! FILE 'VAR%7' MAY NOT BE LARGE ENOUGH TO ACCOMMODATE THE OUTPUT. THE PROGRAM MAY NEED TO BE SEGMENTED OR SECTIONED BEFORE A COMPLETE ANALYSIS CAN BE DONE.
MELD	ALLOWS YOU TO CREATE A PROGRAM OF ALL, OR PART OF OTHER PROGRAMS. IT PUTS THE NEW PROGRAM IN 'STORPL' (YOU MUST CREATE THIS FILE). THIS PROGRAM IS HANDY FOR LIFTING ROUTINES FROM VARIOUS PROGRAMS TO CREATE A NEW PROGRAM, OR REAARANGING OLD PROGRAMS FOR FATER EXECUTION BY MOVING FREQUENTLY USED ROUTINES TO THE FRONT. RESULTING PROGRAMS MAY HAVE OUT-OF-SEQUENCE LINE NUMBERS. RSEQ MUST BE RUN TWICE TO STRAIGHTEN IT OUT WATCH OUT!!! CHECK ALL GOTO'S AND GOSUB'S CAREFULLY!!!
NEWVAR	IS SIMILAR TO MELD, BUT IT ALLOWS YOU TO MERGE 2 OR MORE PROGRAMS WITH COMMON VARIABLE NAMES, AND CHANGE THE VARIABLE NAMES, WHERE DESIRED, TO DIFFERENT NAMES.
DIRCHN	ALLOWS PROGRAM ACCESS RIGHTS TO BE CHANGED ON DISK.
GENLED	A GENERAL LEDGER PROGRAM.

Do you have an unusual use for your computer? An unusual location, an original configuration, some unconventional components? Let the rest of us know about it. Send an article to the OSIO Newsletter.

That little utility program you wrote--isn't it likely that someone else will need to do exactly the same job? Why not send it in to the OSIO Program Exchange?

Watch the label! The expiration date of your membership appears to the right of your membership number. If there's no date, or if the expiration date is near, send money to the treasurer! We don't have the resources to send notices or followups.

More members will give us more resources with which to serve all members better. If you know someone who has an OSI computer and isn't a member of OSIO, do a little missionary work for the organization. Show him or her your Newsletters and explain the advantages of membership.

Cassette Dependability

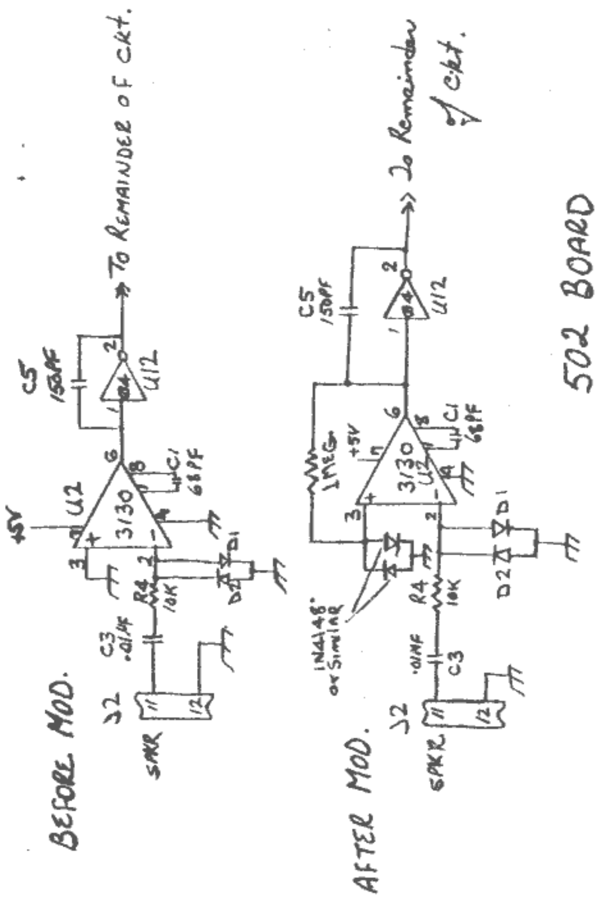
Shans Basiti, of Grand Prairie, Texas, sent in this modification which he worked out for a C4P owner.

"The problem was that every time he wanted to load from his cassette he had to adjust the volume control to exactly the right level to load without an error.

"With the modification shown below we solved the problem. Now he uses a $22.50 cassette player, and all he has to do is turn the volume full open and run it."

"I think this would probably work on any OSI machine with a cassette interface."

There's a Texas saying, "If it ain't broke, don't fix it" -- but if you're having trouble loading without errors, you might want to try this one.



```
 550 P$=" "
 560 POKE530,1:K=57088
 570 D=54095
 580 GOTO620
 590 C=INT(12*RND(8))
 600 D=INT(12*RND(8))
 610 B=C*D:GOTO410
 620 FORT=0TOTK
 630 POKEK,127
 640 P=PEEK(K)
 650 IFP=127THENP$=P$+"1":GOTO820
 660 IFP=191THENP$=P$+"2":GOTO820
 670 IFP=223THENP$=P$+"3":GOTO820
 680 IFP=239THENP$=P$+"4":GOTO820
 690 IFP=247THENP$=P$+"5":GOTO820
 700 IFP=251THENP$=P$+"6":GOTO820
 710 IFP=253THENP$=P$+"7":GOTO820
 720 POKEK,191
 730 P=PEEK(K)
 740 IFP=127THENP$=P$+"8":GOTO820
 750 IFP=191THENP$=P$+"9":GOTO820
 760 IFP=223THENP$=P$+"0":GOTO820
 770 IFP=251THENP$=" ":FORY=DTOYN:
POKEY,32:NEXT
 780 POKEK,223
 790 P=PEEK(K)
 800 IFP=247THEN970
 810 GOTO890
 820 FORY=1TOLEN(P$):POKED+Y,ASC(M
ID$(P$,Y,1)):NEXT
 830 POKED+Y,32:YN=D+Y
 840 FORT1=0TO50
 850 P=PEEK(K)
 860 IFP=255THEN880
 870 NEXT T1
 880 REM
 890 NEXT T
 900 IFP$=B$THENA$=P$:GOTO1010
 910 PRINT"TIME IS UP
 920 PRINT:PRINTQ$;B$
 930 E$(EN)=Q$+B$:EN=EN+1
 940 FORZ=0TO TD:NEXT:X=USR(X)
 950 IFN=PN THEN1100
 960 GOTO320
 970 A$=P$
 980 PRINTA$
 990 P$=" "
 1000 IFA$<>B$THEN1050
 1010 PRINT:PRINT"RIGHT ";Q$;A$
 1020 FORZ=0TO500:NEXT:X=USR(X)
 1030 IFN=PN THEN1100
 1040 GOTO320
 1050 PRINT:PRINT"WRONG":PRINT"THE
 CORRECT ANSWER IS
 1060 E$(EN)=Q$+B$:PRINTE$(EN):EN=
EN+1
 1070 FORZ=0TO TD:NEXT:X=USR(X)
 1080 IFN=PN THEN1100
 1090 GOTO320
 1100 PRINT:PRINT:PRINT:N1=10-EN+E
1
 1110 PRINT"YOU GOT "N1" RIGHT
 1120 PRINT"OUT OF TEN
 1130 IFEN-E1=0THEN1170
 1140 PRINT"YOU MISSED... ":PRINT
 1150 FORX=E1TOEN
 1160 PRINTE$(X):PRINT:NEXT
 1170 IF N=99THEN1210
 1180 INPUT"DO YOU WANT TO CONTINU
E";C$
 1190 E1=EN:PN=PN+10
 1200 IFLEFT$(C$,1)<>"N"THEN320
 1210 X=USR(X)
 1220 PRINT"YOU DID "N " FLASHCARD
S"
 1230 N1=N-EN
 1240 PRINT"AND GOT "N1" RIGHT
 1250 PRINT"YOU MISSED....":PRINT
 1260 FORX=0TOEN
 1270 PRINTE$(X):NEXT
 1280 END
OK
```

Club Discounts

This is not a new offer, but a change in an old one. Richard Comeau, who operated Peripheral Interfacing Equipment of Columbia, MO, has left the field. He's no longer an OSI dealer.

Why? Because he went to work for OSI, that's why.

He'll be helping to set up those Ward's computer stores, and it might not look good for a full-time OSI employee to be selling OSI equipment on the side, and furthermore, OSI will probably have him so busy in the new stores that he wouldn't have time to count the orders anyway. His new address is Richard Comeau, 1640 W. Hwy. 36, Apt. 137, Roseville, MN 55113.

Does that mean no more OSIO discounts from P. I. E.? Well, not exactly. Where does a boy learn about computers? At his father's knee. And Richard's father, Dr. Richard A. Comeau, a computer buff of long standing, will take over P. I. E.

He's already started streamlining procedures, according to Richard. No more long list of specific prices for OSIO members--just a straight 18% discount from list. We hope to have more details next month.

OSI Toolbox

MONEY MODE FIX

Radford Compton writes about a bug in many versions of OS-65U.

"The money mode bug makes itself known by crashing the system when you try to print a very small amount. To test and see if you have a bad version, boot up the system, remove all the floppies from your drives, and then, in immediate mode, run the short program given below. (If you don't remove the floppies, and the system crashes, it might attempt an unauthorized write)."

You never try to print fractions of a cent? How do you know? When you are performing computations, you might very easily encounter a very small figure without even realizing it.

```
PRINT%R,1/104
```

If your system chokes over that mouthful, make the change illustrated below.

```
DISK CHANGE UTILITY

MODE: HEX(H), DEC(D) ? H
UNIT ? A
ADDRESS OFFSET ? C00
ADDRESS ? 25E9
000025E9   F0 ? 20
000025EA   04 ? E2
000025EB   C8 ? 31
000025EC   C8 ? EA
000025ED   C8 ? B0
000025EE   C8 ? 12
000025EF   A2 ? .
ADDRESS ? B0E2
000031E2 J 4A ? F0
000031E3 J 4A ? 04
000031E4   AA ? C8
000031E5 h 68 ? C8
000031E6 ) 29 ? C8
000031E7   0F ? C8
000031E8   18 ? C0
000031E9 i 69 ? 0C
000031EA   0A ? 60
000031EB   CA ? X

OK
```

OSI -- and so, Forth:

Several versions of Forth are now available for OSI machines, and we'd like to hear from Forth users. We have no doubt that we will, because they seem to have an evangelical fervor. Forth users work hard to help the rest of us to see the light.

In this respect they are much like Pascal users. In nearly everything else the two camps are far apart. The differences are illustrated by their attention to the stack: the Pascal programmer never knows what's on the stack, and the Forth programmer never forgets for an instant.

We who are beginners in both languages yearn for enlightenment. Pascal is available, of course, from your OSI dealer. Forth may not be, so we'll give you a few sources, with the understanding that you'll write and tell us how to avoid some of the mistakes you made in learning it. The full Forth Interest Group version (FIG-Forth) is available for OSI systems from (at least) the following vendors:

Technical Products Co., Box 12983, Gainesville, FL 32604 (An earlier version by this firm and Programma did not include the full FIG version. Their OSI-Forth 2.0, they say, does. 8" disks.)

Mike Alport, 5 Woodland Mounds Rd., Iowa City, IA 52240 (We received an announcement saying that the full FIG version is available on 5-1/4" disk for C1P and Superboard. Requires 16K or more memory; $50. In case you're really in a hurry, the telephone number is (319) 338-2883.)

Tailgate Market

Siemens disk drives -- Model FDD220-8. Double-sided, double-density drives, only one year old. Checked out and warranted for 30 days. $375 each or $700 for the pair. Chuck Popenoe, 6307 Wiscasset Rd., Bethesda, MD 20016.

Patches Speed Basic

via Lee Conyers - Reston, Va.

Finally, Byte magazine has published an article that clearly aids OSI users. In the May 1981 issue, pages 236-242, John Sauter of Ann Arbor, Michigan shows how to significantly increase the speed of OSI Basic in performing certain mathematical functions. Impressive improvements are possible in Basic functions using Microsoft's multiplication routine.

John's article is extremely enlightening and I strongly recommend that you read it. He gives complete programs for installing two interpreter patches in both 65D and 65U. One patch is for the multiplication routine, the other for the normalization routine.

Both of John's programs use the same approach for installing the patches in both operating systems even though 65D and 65U are very different. Since I am a 65U user, I preferred the CHANGE utility to install the patches.

To install John's multiplication patch in 65U, using the CHANGE utility, enter the following fifteen HEX values beginning at $1946:

```
66,73,66,74,66,75,66,76,66,BD,98,4A,D0,D6,60
```

To install John's normalization patch in 65U, using the CHANGE utility, enter the following thirteen HEX values beginning at $1854:

```
76,02,76,03,76,04,68,6A,C8,D0,E8,18,60
```

In testing the patches I found a significant improvement in my benchmark program listed below:

```
100 A=9999999
110 PRINT"BEGIN"
120 FORI=1TO5000
130 X=A*A*A
140 NEXTI
150 PRINT"FINISH"
```

Before the patches, the above program took about 28 seconds to execute. After the patches, it took about 20 seconds, for a 28% decrease in execution time. (My 6502A runs at 2 Mhz.) Needless to say, I am delighted.

One final note. Would you believe that when John contacted OSI about his discovery, OSI actually responded? I wonder ...

Membership Information

OSIO dues are $15 a year. Overseas members who want the Newsletter sent by airmail pay an additional $15, a total of $30 U.S. (It seems likely, because of recent increases in postal fees, that we shall have to increase all rates. Renew now!)

If you are a member, your membership number and the date to which your dues are paid are shown above your name on the mailing label. Watch your label, and renew your membership promptly before it expires! We do not have the resources to mail renewal and reminder notices.

If no number and date are shown, this is a sample copy. We invite you to become a member, take part in our activities, and enjoy the benefits of our Newsletter, Program Exchange, chapter meetings, seminars, and occasional discount offers and group purchases. Just send a check in the proper amount to our Treasurer, whose name and address are given below.

Meetings

The MD/VA/DC group meets on the first Tuesday of each month at 7:30 p.m., at the Walter Johnson High School, 6400 Rock Spring Drive, Rockville, MD. On June 2 we will have a review of some popular business programs, in addition to our usual wide-ranging discussion of hardware and software problems and fixes.

OSI discussion groups meet on the second, third, and fourth Tuesday of each month at Community Computers, 2704 N. Pershing Drive, Arlington, VA. On the second Tuesday of each month, discussions will center on cassette-based systems, BASIC-in-ROM, and hardware. On the third Tuesday of each month the subject will be OS-65D, home systems, polled keyboards, and MDMS. On the fourth Tuesday discussions will be about 65U, serial terminals, 8" floppy and hard disks, and business applications.

Program Exchange

Our exchange disks do not contain the operating systems or other proprietary programs; add your own system to them, or copy the programs you want to other disks.

Send in your own programs with enough postage to return the material to you, and if your program is accepted for the Exchange, we'll add the other programs in the Exchange to your disk or cassette and return it.

Primarily for the benefit of new members who do not yet have programs to exchange, we will provide a library disk for $7.50 plus postage, or a cassette for $2.50 plus postage. In the U.S. postage for a disk or a cassette is about $1.80; overseas postage rates have recently been steeply increased. They are now about $2.50 for one 8" disk by air to Central America, South America, or Europe, and about $3 by air to the Pacific Ocean islands and Asia. Each additional disk or cassette adds about .50 more.

In ordering a disk, be sure to say whether 65U or 65D, and give your membership number.

OSIO Officers

President:
John Randall, 10220 Conover Drive,
Silver Spring, MD 20850
Executive Vice-President:
David Morganstein, 13329 Woodruff Place,
Germantown, MD 20767
Treasurer and Corporate Secretary:
Mary Lowery, 2005 N. Uhle St.,
Arlington, VA 22201
Chairman, Program Exchange Committee:
Thomas M. Rowlett, 11296 Windsor Court,
Ijamsville, MD 21754
Corresponding Secretary and
Newsletter Editor: Wallace Kendall,
9002 Dunloggin Rd.,
Ellicott City, MD 21043
Directors:
Sidney H. Brounstein, 9612 McAlpine Rd.,
Silver Spring, MD 20901
William J. Callaghan, 6605 Fisher Ave.,
Falls Church, VA 22046
Joe Kirshner, 9234 E. Parkhill Dr.,
Bethesda, MD 20034
G. D. Marton, 2710 S. Joyce St.,
Arlington, VA 22202
C. H. Popenoe, 6307 Wiscasset Rd.,
Washington, DC 20016

OSIO
MATRIX
6845 Elm Street, Suite 107
McLean, Virginia 22101

BULK RATE
U. S. POSTAGE
PAID
McLEAN, VIRGINIA
PERMIT NO. 178

contains time sensitive material